AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A system for evaluating and selecting programming code, comprising:

a first evaluator, executed by a computer system, to measure ~~a first characteristic~~ power consumption of a plurality of input binaries and compute a plurality of ~~first~~ power consumption figures of merit for the plurality of input binaries, respectively, based on the measured power consumption;~~ and~~

a second evaluator, executed by the computer system, to measure code size of the plurality of input binaries and compute a plurality of code size figures of merit for the plurality of input binaries, respectively, based on the measured code size; and

a binary selector, executed by the computer system, to compute a plurality of overall figures of merit for the plurality of input binaries, respectively, wherein each overall figure of merit is computed as a function of the respective power consumption figure of merit and the respective code size figure of merit of that input binary,

the binary selector to compare the plurality of ~~first figure of merit and~~ overall figures of merit with each other to select one of the plurality of input binaries as having the highest or lowest overall figure of merit.

2. (Currently Amended) The system of claim 1 ~~further comprising:a second evaluator to measure a second characteristic of the plurality of input binaries and compute a plurality of second figures of merit for the plurality of input binaries, respectively, wherein the binary selector is~~ wherein the binary selector is to compare the plurality of power consumption figures of merit with each other and to compare the plurality of ~~second~~ code size figures of merit with each other.

3. (Currently Amended) The system of ~~claim 2 wherein the binary selector is to compute an overall figure of merit for each of the input binaries as a function of the input binary's first and second figures of merit~~ claim 1 wherein the binary selector is to compute the plurality of overall

figures of merit by assigning a first weight to the plurality of power consumption figures of merit and a second weight to the plurality of code size figures of merit.

4. (Currently Amended) The system of claim ~~2~~ 1 further comprising:

a third evaluator, executed by the computer system, to measure ~~a third characteristic~~ performance of the plurality of input binaries and compute a plurality of ~~third~~ performance figures of merit for the plurality of input binaries, respectively, wherein

the binary selector is to compare the plurality of ~~third~~ performance figures of merit with each other.

5. (Currently Amended) The system of claim ~~2~~ 4 wherein ~~the first characteristic is performance,~~ the greater the performance the smaller its associated figure of merit, and ~~the second characteristic is compressed file size,~~ the smaller the ~~compressed file~~ code size the smaller its associated figure of merit.

6. (Currently Amended) The system of claim 2 wherein the binary selector is to compare the plurality of ~~first~~ power consumption figures of merit and ~~second~~ code size figures of merit by computing a mathematical operation for each of the input binaries which includes the respective ~~first~~ power consumption figures of merit and ~~second figures~~ code size figure of merit of that input binary.

7. (Currently Amended) The system of ~~claim 6 wherein the first and second measured characteristics are selected from the group consisting of: performance, code size, power consumption, compressed file size, and memory footprint~~ claim 1 wherein the code size includes at least one of code size, compressed file size, and memory footprint.

8. (Currently Amended) The system of claim 7 further comprising:

a code generator, executed by the computer system, that includes a compiler and a linker to process an output of the compiler and produce the input binaries, wherein

the compiler, executed by the computer system, exposes an optimization control to its user selected from the group consisting of: loop-unrolling; vectorization; and constant propagation.

9. (Original) The system of claim 8 wherein the code generator further comprises a binary rewriter to process an output of the linker and produce the input binaries, wherein the binary rewriter exposes an optimization control to its user selected from the group consisting of: constant propagation; code shrinking; and specialization.

10. (Currently Amended) The system of claim 8 further comprising:

a script processor, executed by the computer system, to process an input script from the user, the script processor to read a plurality of optimization combinations from the input script and configure the code generator in accordance with the optimization combinations, wherein

the code generator is to produce the input binaries as configured by the optimization combinations, respectively.

11. (Currently Amended) The system of claim 7 further comprising:

a binary rewriter, executed by the computer system, to produce the input binaries based on a source binary, wherein

the binary rewriter is to expose optimization controls to its user.

12. (Currently Amended) The system of claim 11 further comprising;

a script processor, executed by the computer system, to process an input script from the user, the script processor to read a plurality of optimization combinations from the input script and configure the binary rewriter in accordance with the optimization combinations, wherein

the binary rewriter is to produce the input binaries as configured by the optimization combinations, respectively.

13. (Currently Amended) A machine-implemented method for processing computer programming code, comprising:

a) producing a current version of a binary using a current optimization setting;

b) measuring ~~a characteristic~~ code size of the current version and computing a current code size figure of merit (FOM) associated with the current version based on the measured code size;

c) measuring performance of the current version and computing a current performance FOM associated with the current version based on the measured performance;

d) computing a current overall FOM as a function of the current code size FOM and the current performance FOM and comparing the current overall FOM with a previously computed overall FOM associated with a prior version of the binary; and

automatically repeating ~~a)-c)~~ a)-d) for another optimization setting.

14. (Currently Amended) The method of claim 13 further comprising:

indicating to a user the version of the binary that has the highest or lowest overall FOM as determined from the comparisons.

15. (Currently Amended) The method of claim 14 further comprising:

ranking a plurality of versions of the binary in accordance with their respective overall FOMs as determined from the comparisons.

16. (Original) The method of claim 13 wherein the current and another optimization settings include optimization controls for compilation, linking, and binary rewriting, and wherein said producing comprises:

compiling source code and linking object files to produce an initial version of the binary, and rewriting the initial version into the current version, using the current optimization setting.

17. (Original) The method of claim 13 wherein the current and another optimization settings include optimization controls for compilation and linking, and wherein said producing comprises:

compiling source code and linking object files to produce the current version of the binary, using the current optimization setting.

18. (Original) The method of claim 13 wherein the current and another optimization settings include optimization controls for binary rewriting, and wherein said producing comprises:

rewriting an initial version of the binary into the current version, using the current optimization setting.

19. (Currently Amended) The method of claim 13 further comprising:

~~d) measuring another characteristic of the current version and computing another figure of merit (FOM) associated with said another characteristic and the current version; and~~

e) comparing the current code size FOM with a previously computed code size FOM that is associated with code size and with the prior version of the binary; and

f) comparing ~~said another~~ the current performance FOM with a previously computed performance FOM that is associated with ~~said another characteristic~~ the performance and with ~~a~~ the prior version of the binary.

20. (Currently Amended) The method of claim 13 wherein the binary comprises a firmware driver, and ~~the characteristic is compressed file size of the binary~~ the code size includes at least one of code size, compressed file size, and memory footprint.

21. (Currently Amended) An article of manufacture comprising:

a non-transitory machine-accessible storage medium containing instructions that, when executed, cause a machine to:

a) generate a binary under an optimization setting;

b) compute a performance cost as a function of a measured ~~characteristic~~ performance of the binary;

c) compute a power consumption cost as a function of a measured power consumption of the binary,

d) compute an overall cost for the binary as a function of the performance cost and the power consumption cost;

e) perform ~~a) – b)~~ a)-d) a plurality of times each time with a different optimization setting but based on the same source program; and

f)~~d)~~ compare the computed overall costs with each other, to select the binary having the lowest overall cost.

22. (Currently Amended) The article of manufacture of claim 21 wherein the instructions cause the machine to perform ~~a) – b)~~ a)-d) a plurality of times, by first compiling the source program and then rewriting the binary a plurality of times and then recompiling the source program and then rewriting the recompiled binary a plurality of times.

23. (Currently Amended) The article of manufacture of claim 21 further comprising instructions that cause the machine to perform ~~b)~~ c) by computing a ~~further~~ code size cost as a function of a measured~~, further characteristic~~ code size of the binary generated in a).

24. (Currently Amended) The article of manufacture of claim 23 wherein the instructions cause the machine to compare the computed overall costs in ~~d)~~ f), by computing an overall cost for each generated binary, wherein the overall cost is a function of ~~said cost and said further cost computed in b)~~ the performance cost, the power consumption cost, and the code size cost.